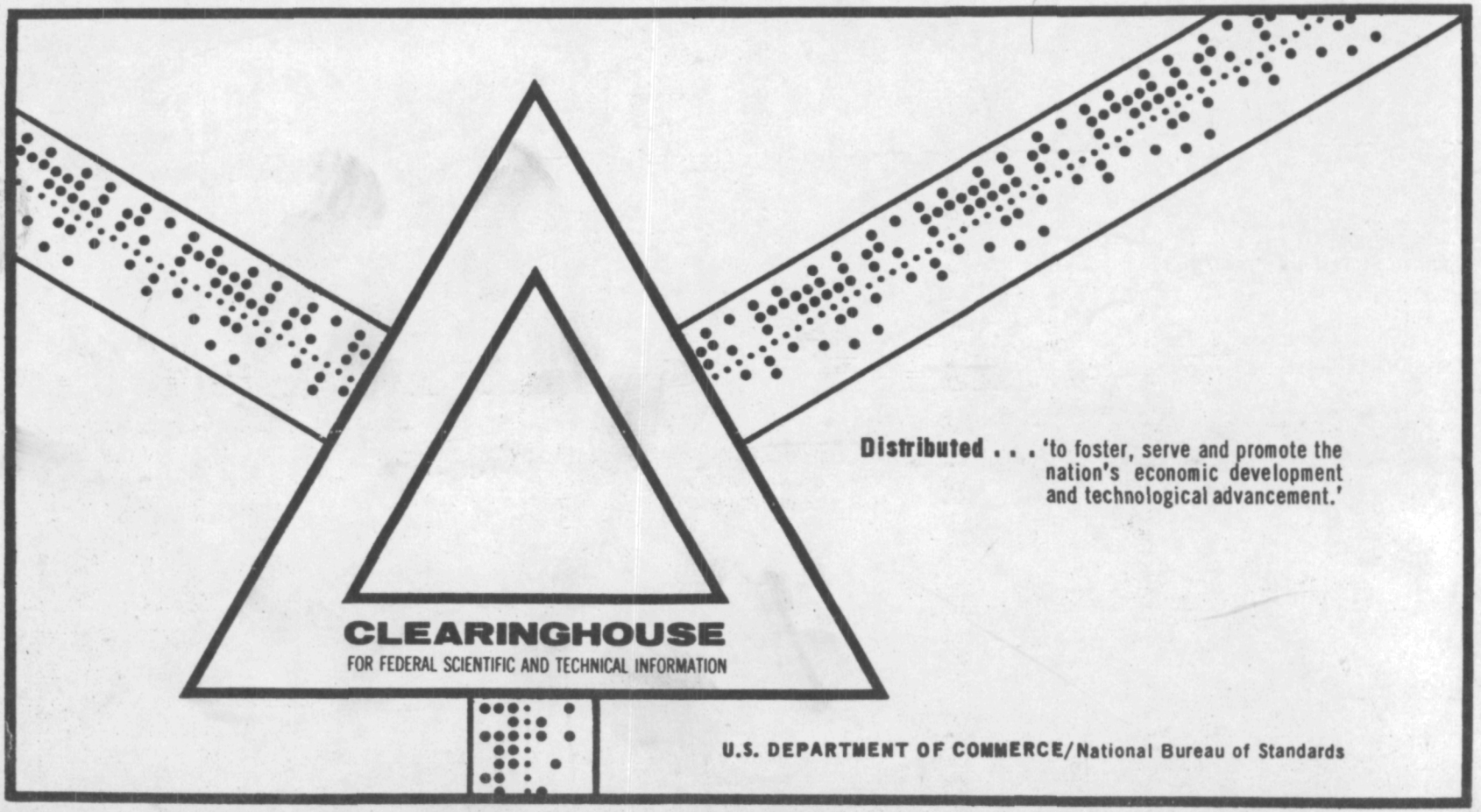
NASA CR-96005

TWO NEW METHODS FOR OBTAINING STABILITY DERIVATIVES FROM FLIGHT TEST DATA

George H. Burgin

Decision Science, Inc.
San Diego, California

September 1968



NASA CR-96005 -REV
ii

TWO NEW METHODS FOR OBTAINING STABILITY DERIVATIVES FROM FLIGHT TEST DATA

By George H. Burgin

September 1968

Prepared under Contract No. NAS4-1280 by
DECISION SCIENCE, INC.
San Diego, California

Flight Research Center
NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

ABSTRACT

Two new methods for obtaining control and stability derivatives from observed flight data are developed. The first method is based on a quasilinearization procedure and is applicable in parameter identification problems where the plant is modeled by a system of linear differential equations, and noisy measurements of state and control variables are available. Computationally, this method is equivalent to a modification of the Newton-Raphson method. The second, a "directed random search" method is based on a concept called evolutionary programming, and is also applicable for nonlinear problems. Using X-15 flight test data, the two methods are compared and stability and control derivatives for the lateral motion of the X-15 are given.

FOREWORD

This research on computational methods for calculating stability derivatives from observed flight test data was sponsored by the National Aeronautics and Space Administration Flight Research Center in Edwards, California, under Contract No. NAS4-1280. The NASA project monitors were Mr. Lawrence W. Taylor, Jr. and Mrs. Harriet J. Smith.

These studies were performed by Decision Science, Inc., San Diego, in the period December 1967 through September 1968. The principle investigator was Dr. George H. Burgin. Dr. M. J. Walsh served as a technical advisor throughout this research. Mr. George Kurata wrote most of the digital computer programs developed under this contract.

CONTENTS

LIST OF FIGURES

TWO NEW METHODS FOR OBTAINING STABILITY DERIVATIVES FROM FLIGHT TEST DATA

By George H Burgin
Decision Science, Inc

SUMMARY

Two new methods for calculating stability and control derivatives from flight test data are developed Digital computer programs for both methods have been written and tested out with actual flight test data delivered by NASA Flight Research Center, Edwards

The first method is applicable whenever the system is represented by a set of linear differential equations and the error criterion is that of minimizing the squared error integral It uses parameter sensitivity functions (gradients) to obtain an algorithm which minimizes the error function This method shows very good convergence and can be extended to certain nonlinear differential equations It is shown that a modification of the Newton-Raphson method will result in the same computational algorithm

The second method is a direct search method in the space of the unknown system parameters The search proceeds by changing one parameter at a time, progressing stepwise to points in the parameter space which yield lower and lower error function values The most likely successful next step is determined by a finite-state machine, which, in itself, is obtained by a search procedure called evolutionary programming

Obtaining stability derivatives from flight test data is a special case of the process parameter identification problem and modifications of these two methods are applicable whenever the problem of obtaining system parameters from measured (and therefore noisy) state and control variables has to be solved

The analysis of flight test data of an X-15 flight shows that the first method is computationally more efficient than the second one Evolutionary programming may have advantages in the case of process identification problems dealing with nonlinear differential equations or with nonquadratic error functions The first method permits obtaining error estimates for the unknown parameters and the experiments with the given X-15 flight test data indicate that these error estimates are of realistic size

It is suggested that these methods be extended to attack the problem of obtaining stability derivatives of airplanes with non-negligible nonlinear properties

SYMBOLS

A		coefficient matrix for the state variables (stability matrix)
a_{ij}		element of A
B		coefficient matrix for the control variables (control matrix)
b_{ij}		element of B
C		inverse of the normal equations
L_p	=	$\frac{1}{I_x}\left(\frac{\partial L}{\partial p}\right)$ dimensional stability derivative parameter
$L_{\delta A}$	=	$\frac{1}{Ix}\left(\frac{\partial L}{\partial \delta_A}\right)$ dimensional control derivative parameter
L_r, L_β, $L_{\delta R}$, N_p, N_r, N_β, $N_{\delta A}$, Y_β, Y_ϕ		dimensional stability or control derivative parameters analoguos to the preceding definition
F		linear vector function for the derivatives of the state variables
H_j	=	$\frac{\partial z}{\partial \alpha j}$ partial derivative of the cost function with respect to the j-th unknown parameter
M		(M+1) = number of observed data points
m		number of control variables
n		number of state variables
p		roll rate, radians/second
r		yaw rate, radians/second
t		time, seconds

SYMBOLS (concluded)

T	observation period, seconds
U	total number of unknown systems parameters
u	control variable
w	weighting factor
x	state variable, calculated
$\tilde{x}$	state variable, observed
z	cost function
α	angle of attack, radians
α_j	j-th unknown system's parameter
β	sideslip angle, radians
δ	(in connection with evolutionary programming change in the cost function)
δA	aileron deflection, radians
δR	rudder deflection, radians
δo	dummy control variable
δ_{ik}	Kronecker delta
σ	standard deviation
ϕ	bank angle, radians

Superscripts

T	transpose of vector or matrix
o	nominal condition, reference trajectory
k	trajectory obtained in k-th iteration

a dot denotes the time derivative

INTRODUCTION

Determination of stability and control derivatives from flight test data is a special problem from the more general area of systems identification, a field which has received a great deal of consideration during the last few years A short summary and additional references to earlier methods of determining stability derivatives from flight test data can be found in a recent paper by Lawrence Taylor (ref 1) and in an article by Peter Young (ref 2)

The work described in this report was performed under contract NAS4-1280 by Decision Science, Inc in San Diego for the NASA Flight Research Center at Edwards, California The purpose of this contract was to use a new computer technique, called evolutionary programming (ref 3) for the determination of stability derivatives and to compare the computational efficiency of this method with the efficiency of other, more analytical, methods The analysis of the possible mathematical techniques showed that a very efficient computational algorithm can be derived by either using a quasilinearization technique or a modification of the classical Newton-Raphson method In addition to its efficiency, this method permits estimates of the variances of the calculated stability and control derivatives.

Both methods have been implemented in digital computer programs (for a CDC 3600 and an IBM 360/40 computer) and observed X-15 flight data have been analysed The assistance of Lawrence Taylor and Harriet Smith, both of NASA Flight Research Center, in conducting this work is greatly appreciated

Statement of the Problem

In the problem of parameter identification of a linear system, the process is assumed to be governed by the linear matrix differential equation

$$\dot{x} = Ax + Bu \qquad (1)$$

where x is a vector of n state variables and u a vector of m control variables, A a coefficient matrix (stability derivatives) with dimension $(n \times n)$ and B a coefficient matrix (control derivatives) with dimension $(n \times m)$ It is assumed that some or all of the elements of

A and B are unknown and to be determined. Unless otherwise stated, these elements will be assumed to be time invariant

In order to obtain estimates of the elements of A and B, measured time histories of the state variables x and the control variables u, and possibly of their time derivatives, are given The problem consists now in finding a_{ij}'s and b_{ij}'s which will, when substituted into the differential equations, match the observed time histories if the differential equations are solved with the correct initial conditions and the measured control variables $u(t)$

In the case of the determination of stability and control derivatives from flight data, it can be assumed that the time histories have been obtained in carefully planned experiments The accuracy and the confidence limits of the calculated parameters depend a great deal on the proper choice of the forcing functions, which are, for the lateral motion, rudder and aileron deflection These two functions not only have to be linearly independent but must have an amplitude which compromises between a large value (therefore obtaining a good signal to noise ratio in the measured data) and a small value (therefore obtaining a motion with negligible nonlinear effects) (ref 4)

If a time history is generated by a process which is governed exactly by a linear differential equation of form (1), and if no noise or errors are introduced in the measurement of $x(t)$, $\dot{x}(t)$ and $u(t)$, the $n(n + m)$ unknowns can be found by formulating $n(n + m)$ linearly independent equations using observed values at $(n + m)$ different time points, and then solving this exact system of linear equations for the unknown parameters

If values for the derivatives of the state variables are not obtainable by measurement, yet the state variables themselves are measured with sufficient accuracy, numerical differentiation of the state variables gives estimates of the $\dot{x}(t)$ and it is, in principle, again possible to formulate as many linear equations as there are unknowns.

However, since the differentiation is a process which introduces noise, more reliable results can be obtained by setting up more linear equations than there are unknowns and then to solve these equations by a least square procedure. A comprehensive summary of least squares methods can be found in reference 5, and a short summary of the computational procedure for obtaining least squares estimates of the

stability and control derivatives is given in the next section of this report. The least squares technique is used in both the methods described here to obtain a set of starting values for the unknown parameters.

So far, it has been assumed that time histories obtained from processes which can be exactly described by a linear system of differential equations of form (1) are analysed In most practical situations, however, equation (1) is only a first order approximation to a really nonlinear process This, of course, is exactly the case if the lateral motion of an airplane is approximated by a system of linear differential equations Already the existence of purely lateral motion is a simplifying assumption, the exact form of the equations of motion for airplanes shows that there is coupling between the lateral and longitudinal motion, as can be seen for instance in *reference 6. Purely lateral* motion described by a linear system of differential equations also neglects the product terms between roll and yaw and the nonlinear aerodynamic forces produced by the control surface deflections.

An attempt to estimate the error bounds on the calculated stability and control derivatives must take into account the errors introduced by noisy measurement and the error in using a linear model of a nonlinear process

The Linearized Equations of Lateral Motion

The following system of differential equations is used in determining stability and control derivatives for the lateral motion (reference 10)

$$\begin{aligned}
\dot{p} &= L_p\, p + L_r\, r + L_\beta\, \beta + L_{\delta A}\, \delta A + L_{\delta R}\, \delta R + L_o \\
\dot{r} &= N_p\, p + N_r\, r + N_\beta\, \beta + N_{\delta A}\, \delta A + N_{\delta R}\, \delta R + N_o \\
\dot{\beta} &= \alpha\, p - r + Y_\beta\, \beta + Y_\phi\, \phi + Y_o \\
\dot{\phi} &= p
\end{aligned} \qquad (2)$$

The last column of constants are multiplied by a constant control force of constant magnitude one These "dummy derivatives" allow for compensation of drift of the null point of the measuring devices

The A and B matrices follow

A

L_p	L_r	L_β	0
N_p	N_r	N_β	0
α	-1	Y_β	Y_ϕ
1	0	0	0

B

$L_{\delta A}$	$L_{\delta R}$	L_o
$N_{\delta A}$	$N_{\delta R}$	N_o
0	0	Y_o
0	0	0

The element a_{31} represents the angle of attack of the airplane during the flight test When performing a lateral maneuver, the pilot attempts to keep the angle of attack constant Since α can be measured, it is not considered as one of the unknowns of the elements of the A matrix, in the calculation, the actually (and slightly varying) measured value of the angle of attack is used

The value of Y_o is also measured and therefore a known quantity so that there are 14 unknowns to be determined

THE LEAST SQUARES METHOD

The least squares method is based on the fact that the differential equations (2) are valid at every time instant. Substituting measured values of p, r, β and ϕ and of their derivatives into the differential equations permits forming simultaneous linear algebraic equations. Setting up more linear equations than there are unknowns yields an overdetermined system of linear equations, the solution of which gives the estimates of the unknown parameters.

In the general case where the model of the system is given by a differential equation of form (1), the following four steps are required to obtain least square estimates.

Step 1. In this preliminary step, obtain numerical approximations to those derivatives of the state variables which are not available from measurements. If the observed values of the state variables are not too contaminated by noise, the following approximation can be sufficient.

$$\tilde{\dot{x}}_i(t_k) \approx \left[\tilde{x}_i(t_{k+1}) - \tilde{x}_i(t_{k-1})\right] / (t_{k+1} - t_{k-1})$$

If the observed values are unequally spaced or are very noisy, higher order approximations may have to be used. Good results can be obtained by smoothing and interpolation with spline functions (ref 7).

Step 2. Number the unknowns in consecutive order, for instance

$$a_{11} = \alpha_1$$
$$a_{1i} = \alpha_i$$
$$\vdots$$
$$a_{nn} = \alpha_{(n*n)}$$
$$b_{11} = \alpha_{(n*n)+1}$$
$$b_{nm} = \alpha_{(n*n)+(n*m)}$$

Step 3. Each time instant, at which measured values of the state variables $\tilde{x}_i$ $i=1 \ldots n$ and the control variables u_j $i=1 \ldots m$ and values of the derivatives of the state variables $\tilde{\dot{x}}_i$ $i=1 \ldots n$ are available allows (by substitution into the differential equations) the formulation of n linear equations, in which the $\tilde{\dot{x}}_i$ are the coefficients of the unknown parameters α_k, $k=1 \ldots n*n$, the u_j are the coefficients of the unknowns α_k, $k=(n*n)+1 \ldots n*(n+m)$. The values of the derivatives of the state variables form the right hand sides of these equations. The form of these equations is illustrated for a specific case with three state variables and one control variable in equation 3. It is clear from the form of these equations that they can be separated into n independent systems, one separate system for every state variable, and therefore for the unknown parameters of one row of the A and B matrix.

Step 4. The overdetermined system can now be solved by the classical least squares method and the estimates of the unknown coefficients as well as their variances can be determined.

If some of the elements of the A or B matrix are assumed to be known, the overdetermined system of linear equations is obtained by subtracting the products of the given parameters with their corresponding coefficients from the right hand side.

In order to show how the variances are calculated, write the overdetermined system of equations in the form

$$A\,\alpha = b \qquad (4)$$

(one such system for each state variable)

Assume a vector α', for which

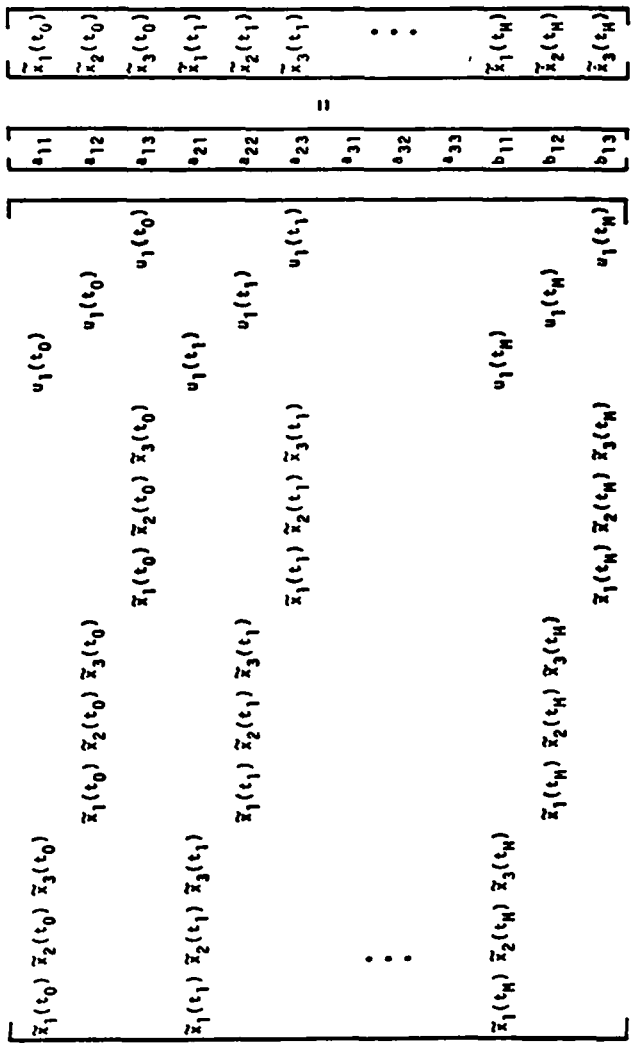
$$A\,\alpha' = c$$

and denote the difference vector between b and c with v

$$v = b - c$$

The least squares solution requires $v^T v$ = minimum and it is obtained by solving

$$A^T A \alpha' = A^T b \qquad (5)$$



(3)

Equations (5) are called the normal equations, and their solution can be expressed as

$$a^{\circ} = (A^T A)^{-1} A^T b \tag{6}$$

The variance of the total fit error is given by

$$\sigma^2 = \frac{v^T v}{(M+1)-n} \tag{7}$$

where M+1 is the total number of equations (see eq (3)) and n is the number of unknowns. The quantity M+1-n is called the degree of freedom

The standard deviation for the i-th unknown parameter can be calculated as

$$\sigma_{a_i} = \sigma \sqrt{c_{ii}} \tag{8}$$

where c_{ii} is the i-th main diagonal element of the inverse of the normal equation

$$C = (A^T A)^{-1}$$

The value of $v^T v$ can be obtained in two ways Obviously, by backsubstituting the solutions found for the a° into equation (4), then summing the squares of the residuals $(b_i - c_i)^2$ yields the scalar quantity $v^T v$ Computationally more efficient is the following way

$$v^T v = b^T b - (A^T b)\, a^{\circ}$$

The correctness of this expression can be seen as follows

$$\begin{aligned} v^T v &= (Aa' - b)^T (Aa^{\circ} - b) \\ &= (a'^T A^T - b^T)(Aa^{\circ} - b) \\ &= a'^T (A^T A\, a' - A^T b) \quad b^T A a^{\circ} + b^T b \end{aligned}$$

The first term vanishes due to the normal equations (5) and therefore

$$v^T v \quad b^T b - (A^T b)^T a \tag{9}$$

Since the elements $A^T b$ are already known as the right hand sides of the normal equations, this calculation requires only the calculation of two scalar products

Finally, a remark about linear dependencies seems in order Consider first linear dependencies between individual equations (row dependencies) As long as the total number of equations minus the number of linear dependencies is greater than the number of unknowns, the linear dependencies are irrelevant for obtaining parameter estimates Consider now linear dependencies between state or control variables (column dependencies) If the mathematical model expressed by equation (1) is adequate, linear dependencies between the state variables are impossible However, linear dependencies between control variables are possible and if two or more control variables are linearly dependent over the entire period of observation, two or more columns in the overdetermined system, and therefore also in the normal equations will be linearly dependent In this case, it is not possible to determine all the unknown parameters in the B matrix, only ratios between parameters can be calculated

In a process identification problem, where the time histories are obtained by performing carefully planned experiments, linear dependencies between control variables can always be avoided For the determination of the stability and control derivatives of the lateral motion of an airplane, linearly independent aileron and rudder deflections will guarantee a nonsingular coefficient matrix of the normal equations

THE METHOD OF QUASILINEARIZATION

The method developed in this section results in an algorithm which is computationally equal to the one described by Taylor in his paper "A Modified Newton-Raphson Method for Determining Stability Derivatives From Flight Data" (ref 1) It is interesting to note that the same procedure for calculating stability derivatives can be obtained by two quite different approaches

The basic idea is to find coefficients in the A and B matrix which permit fitting the observed time histories (which are assumed to be solutions to the differential equation (1)) As criterion of fit the integral of the weighted squared differences between calculated and observed time histories is chosen The following cost function is therefore defined

$$z = \sum_{i=1}^{n} w_i \int_0^T \left[x_i(t) - \tilde{x}_i(t) \right]^2 dt \qquad (10)$$

If reliable measurements of the derivatives of the state variables are available, a different possible cost function could be taken as

$$z = \sum_{i=1}^{n} w_i \int_0^T \left[x_i(t) - \tilde{x}_i(t) \right]^2 dt + \sum_{i=1}^{n} w_i \int_0^T \left[\dot{x}_i(t) - \dot{\tilde{x}}_i(t) \right]^2 dt \qquad (11)$$

It is also possible to include only certain derivatives in the second sum, eg p and r It seems that the choice of the cost function deserves additional attention

As weighting factors the inverse of the root mean square of the observed state variable gives a reasonable balance between the four observed state variables Different choices for the weighting factors are, of course, possible and may take the relative accuracies of the measurements into consideration (for instance, roll and yaw rates can often be measured more accurately than bank and sideslip angle)

In the following derivation, a cost function of form (13) is assumed and for simplicity of notation, the weight factors are all assumed to be one

As a preliminary step in deriving the algorithm it is shown how the sensitivity functions (here the partial derivatives of the state variables with respect to the unknown parameters) can be calculated Rewrite equation (1) in the following form

$$\dot{x} = F(x, u, \alpha, t) \tag{12}$$

where F is a vector function and all unknown parameters are combined into a row vector α, such that

$$a_{11} \quad a_{1n} \rightarrow \alpha_1 \quad \alpha_n$$
$$a_{21} \quad a_{2n} \rightarrow a_{n+1} \quad \alpha_{2n} \quad \text{etc}$$

Differentiate equation (12) with respect to some α, say α_j, and write the i-th component of the vector (α assumed to be independent of time)

$$\frac{\partial \dot{x}_i}{\partial \alpha_j} = \frac{\partial}{\partial \alpha_j} F_i (x, u, \alpha, t)$$
$$= \sum_{k=1}^{n} \frac{\partial F_i}{\partial x_k} \frac{\partial x_k}{\partial \alpha_j} + \sum_{k=1}^{m} \frac{\partial F_i}{\partial u_k} \frac{\partial u_k}{\partial \alpha_j}$$
$$+ \frac{\partial F_i}{\partial \alpha_j} \tag{13}$$

Since the control variables are independent of α, the second sum vanishes Under the usual assumption that the second partial derivatives are continuous, we can interchange the order of differentiation and obtain

$$\frac{d}{dt}\left(\frac{\partial x_i}{\partial \alpha_j}\right) = \sum_{k=1}^{n} \frac{\partial F_i}{\partial x_k} \frac{\partial x}{\partial \alpha} + \frac{\partial F_i}{\partial \alpha_j} \tag{14}$$

Equation (14) is a linear differential equation for the influence coefficient

$$\frac{\partial x_i}{\partial \alpha_j}$$

Specifically, for a system of form (1) with four state variables and three control variables, the sensitivity equations can be written as

$$\frac{d}{dt}\left(\frac{\partial x_i}{\partial a_{kj}}\right) = \sum_{\nu=1}^{4} a_{i\nu} \frac{\partial x_\nu}{\partial a_{kj}} + x_j \delta_{ik} \qquad \begin{matrix} i=1 & 4 \\ k=1 & 4 \\ j=1 & 4 \end{matrix}$$

$$\frac{d}{dt}\left(\frac{\partial x_i}{\partial b_{kj}}\right) = \sum_{\nu=1}^{4} a_{i\nu} \frac{\partial x_\nu}{\partial b_{kj}} + u_j \delta_{ik} \qquad \begin{matrix} i=1 & .4 \\ k=1 & .4 \\ j=1 & .3 \end{matrix} \tag{15}$$

where δ_{ik} is the Kronecker delta

There are 108 linear differential equations, which can be solved simultaneously with the four equations (1). Therefore, a system of 112 differential equations is obtained The initial conditions of the influence coefficients

$$\frac{\partial x_i}{\partial a_{kj}} \text{ and } \frac{\partial x_i}{\partial b_{kj}}$$

are all zero because the parameters are independent of the initial conditions of the state and control variables Equation (15) indicates how the sensitivity functions are obtained as solutions of a linear system of differential equations

It is now shown how these influence functions can be used to obtain corrections to the unknown coefficients Obtain a first approximation to the unknown (constant) coefficients by applying the least squares technique as described in the preceding section Then solve the differential equations for the state variables together with the equations for the paramter influence coefficients

At each point at which observed values of the state variables are available, expand around the reference point obtained with the present values of the coefficients the solution of the state variable in a Taylor series as a function of the unknown parameter corrections, i e.,

$$x_i(\alpha + \Delta\alpha, t) = x_i^0(\alpha,t) + \sum_{j=1}^{n(n+m)} \frac{\partial x_i}{\partial \alpha_j}(\alpha,t)\Delta\alpha_j \qquad (16)$$

+ higher order terms

In the above expression, consider $x_i(\alpha+\Delta\alpha,t)$ as the desired value (equal to the observed value of $\tilde{x}_i(t)$, $x_i^0(\alpha,t)$ the value of the presently computed reference solution and the summation as the desired correction This clearly gives a linear equation for the $\Delta\alpha_j$. For each state variable, and for each point t, one such equation is obtained When the reference solution is carried out over the entire observation interval, n(M+1) linear equations can be formulated and solved by a least square method This solution yields corrections to the present values of the parameters Adding these corrections to the parameters will give a new reference trajectory, closer to the one of the observed data This process can be repeated until the corrections become negligible.

One way of looking at the problem of obtaining corrections $\Delta\alpha_j$ requires that the n integrands in

$$z = \sum_{i=1}^{n} \int_0^T \left[x_i(\alpha,t) + \sum_{j=1}^{U} \frac{\partial x_i}{\partial \alpha_j}(\alpha,t)\Delta\alpha_j - \tilde{x}_i(t)\right]^2 dt \qquad (17)$$

(U is the total number of unknowns)

vanish In other words, try to satisfy the following equation

$$\sum_{j=1}^{U} \frac{\partial x_i}{\partial \alpha_j}(\alpha,t)\Delta\alpha_j = \tilde{x}_i - x_i^0(\alpha,t) \qquad i=1 \quad 4 \qquad (17a)$$

Formulating these equations for $t=t_0, t_1, \ldots, t_m$ yields an overdetermined system for which the normal equations are of the form

and an IBM 360/40 computer Four to five iterations were required to get the corrections to about 1/10,000 of the value of the coefficient Using approximately 120 time points required approximately one minute computer time on the CDC 3600 to calculate 15 unknown parameters with 5 iterations The differential equations are solved by a fourth order Runge Kutta method

Estimates of the variance of the unknown parameters are obtained in the following way

The variance of the total fit error can be expressed as

$$\sigma^2 = \sum_{i=1}^{n} w_i \sum_{\nu=0}^{M} \left[x_i(t_\nu) - \tilde{x}_i(t_\nu)\right]^2/(M+1-U) \qquad (19)$$

and estimates of the variances of the individual parameters are calculated as

$$\sigma_{\alpha_j} = \sigma^2 c_{jj} \qquad (20)$$

where c_{jj} is the j-th main diagonal element of the inverse of the normal equations (18)

$$\sum_{i=1}^{4}\sum_{\nu=0}^{M}\frac{\partial x_i}{\partial \alpha_1}^2(t_\nu)\Delta\alpha_1 + \sum_{i=1}^{4}\sum_{\nu=0}^{M}\frac{\partial x_i}{\partial \alpha_2}(t_\nu)\frac{\partial x_1}{\partial \alpha_1}(t_\nu)\Delta\alpha_2 +$$

$$= \sum_{i=1}^{4}\sum_{\nu=0}^{M}\left[\tilde{x}_i(t_\nu)-x_i(t_\nu)\right]\frac{\partial x_i}{\partial \alpha_1}(t_\nu)$$

$$\sum_{i=1}^{4}\sum_{\nu=0}^{M}\frac{\partial x_1}{\partial \alpha_2}(t_\nu)\frac{\partial x_i}{\partial \alpha_1}(t_\nu)\Delta\alpha_1 + \sum_{i=1}^{4}\sum_{\nu=0}^{M}\frac{\partial x_1}{\partial \alpha_2}(t_\nu)^2\Delta\alpha_2 + \cdots$$

$$= \sum_{i=1}^{4}\sum_{\nu=0}^{M}\left[\tilde{x}_i(t_\nu)-x_i(t_\nu)\right]\frac{\partial x_i}{\partial \alpha_2}(t_\nu)$$

$$\vdots$$

$$\sum_{i=1}^{4}\sum_{\nu=0}^{M}\frac{\partial x_1}{\partial \alpha_U}(t_\nu)\frac{\partial x_i}{\partial \alpha_1}(t_\nu)\Delta\alpha_1 + \sum_{i=1}^{4}\sum_{\nu=0}^{M}\frac{\partial x_1}{\partial \alpha_U}(t_\nu)\frac{\partial x_1}{\partial \alpha_2}(t_\nu)\Delta\alpha_2 +$$

$$= \sum_{i=1}^{4}\sum_{\nu=0}^{M}\left[\tilde{x}_i(t_\nu)-x_i(t_\nu)\right]\frac{\partial x_1}{\partial \alpha_U}(t_\nu) \qquad (18)$$

The solution to these normal equations yields the corrections $\Delta\alpha_j$, which are added to the old values of the α_j and then a solution of the differential equations for the state variables and sensitivity functions using these new parameters is performed. This process is iterated until the corrections become negligible.

A computer program which allows up to 15 parameters assumed to be unknown was written and run both on a CDC 3600

NEWTON-RAPHSON'S METHOD AND ITS MODIFICATION

This section shows how the same computational algorithm, which is given in equation (18) can be obtained by a modification of the Newton-Raphson method and that it might be worthwhile to program the unmodified Newton-Raphson method. After a short general exposition of how the Newton-Raphson method can be used in optimization problems, it is shown how the necessary partial second derivatives of the state variables with respect to the unknown parameters can be calculated in a manner similar to the one applied for obtaining the first order sensitivity coefficients. It is then shown how the second order partial derivatives of the cost function with respect to the unknown parameters are obtained and used to minimize the cost function.

First, consider the problem of minimizing a function of several independent variables with no constraints. Let

$$z = F(\alpha_1 \quad \alpha_n) = F(\bar{\alpha}) \qquad (21)$$

where $\bar{\alpha}$ denotes the vector with elements $\alpha_1, \quad \alpha_U$

A necessary condition that z has a local minimum is

$$\frac{\partial z}{\partial \alpha_j} = 0 \quad \text{for all} \quad j$$

Since Newton's method is really a procedure to find zeros (not extrema) of functions, it is used to find values of α which will satisfy the above condition. Let

$$H_j(\bar{\alpha}) = \frac{\partial z}{\partial \alpha_j}$$

and expand $H_j(\bar{\alpha})$ around some point $\bar{\alpha}^\circ$ into a Taylor series

$$H_j(\bar{\alpha} + \Delta\bar{\alpha}) - H_j(\bar{\alpha}^\circ) = \sum_{k=1}^{U}\frac{\partial H_j}{\partial \alpha_k}(\bar{\alpha}^\circ)\Delta\alpha_k$$

$$+ \text{higher order terms} \qquad (22)$$

Neglecting the higher order terms and requiring that

$$H_j(\bar{\alpha}^\circ + \Delta\bar{\alpha}) = 0$$

yields a system of U linear equations for the U unknowns

$$\sum_{k=1}^{U} \frac{\partial H_j}{\partial \alpha_k}(\bar{\alpha}^\circ)\Delta\alpha_k = -H_j(\bar{\alpha}^\circ) \qquad j=1 \quad U \qquad (23)$$

Newton's method consists in solving the above system of linear equations. Since the higher order terms have been neglected, $H_j(\bar{\alpha}^\circ + \Delta\alpha)$ will not be exactly zero but, if the starting point was close enough to the zero of H_j, $|H_j(\bar{\alpha} + \Delta\alpha)|$ will be smaller than $|H_j(\bar{\alpha}^\circ)|$. The process is repeated and converges to the zero of H_j with quadratic convergence. Remembering now that H_j is the first partial derivative of the function to be minimized with respect to the j-th unknown parameter, the elements in the coefficient matrix in the above equation are the second partial derivatives of the function to be minimized with respect to the unknown parameters. In order to obtain the second order partial derivatives of the cost function, consider first the procedure to obtain second partial derivatives of the state variables. Write the differential equation which governs the state variables in the following form

$$\dot{x}_i = G_i\left(x(\bar{\alpha}), \alpha, u, t\right)$$

The (vector) function G may be nonlinear. Then

$$\frac{\partial}{\partial\alpha_j}\left(\frac{dx_i}{dt}\right) = \sum_{k=1}^{n} \frac{\partial G_i}{\partial x_k}\frac{\partial x_k}{\partial \alpha_j} + \sum_{k=1}^{m} \frac{\partial G_i}{\partial u_k}\frac{\partial u_k}{\partial \alpha_j} + \frac{\partial G_i}{\partial t}\frac{\partial t}{\partial \alpha_j} + \frac{\partial G_i}{\partial \alpha_j} \qquad (24)$$

Assuming the control variables independent of α, the second term vanishes and since the α are assumed to be time independent, the third term drops out. Interchanging the order of differentiation, we obtain the well known result

$$\frac{d}{dt}\left(\frac{\partial x_i}{\partial \alpha_j}\right) = \sum_{k=1}^{n} \frac{\partial G_i}{\partial x_k}\frac{\partial x_k}{\partial \alpha_j} + \frac{\partial G_i}{\partial \alpha_j}$$

Differentiating again and immediately changing the order of differentiation on the left-hand side gives

$$\frac{d}{dt}\left(\frac{\partial^2 x_i}{\partial\alpha_j\partial\alpha_\ell}\right) = \sum_{k=1}^{n}\left\{\sum_{\mu=1}^{n} \frac{\partial^2 G_i}{\partial x_\mu \partial x_k}\frac{\partial x_\mu}{\partial \alpha_\ell}\frac{\partial x_k}{\partial \alpha_j} + \frac{\partial^2 G_i}{\partial \alpha_\ell \partial x_k}\frac{\partial x_k}{\partial \alpha_j} + \frac{\partial^2 x_k}{\partial \alpha_j \partial \alpha_\ell}\frac{\partial G_i}{\partial x_k} + \frac{\partial^2 G_i}{\partial \alpha_j \partial x_k}\frac{\partial x_k}{\partial \alpha_\ell}\right\} + \frac{\partial^2 G_i}{\partial \alpha_j \partial \alpha_\ell} \qquad (25)$$

This system of differential equations, together with the differential equations for the state variables and the first order sensitivity functions allows the calculation of the second order partial derivatives. It may be emphasized again that the above derivation made no assumption about linearity of the functions $G_i(\bar{\alpha}, \bar{x}, \bar{u}, t)$.

The total number of second partial derivatives and therefore of differential equations of form (25) required in the Newton-Raphson method is nU^2.

For the special case where the functions G_i are linear and of the form

$$G_i(x, \bar{\alpha}) = \sum_{j=1}^{n} a_{ij}x_j(t) + \sum_{j=1}^{m} b_{ij}u_j(t)$$

several simplifications can be made

$$\frac{\partial^2 G_i}{\partial x_s \partial x_t} = 0 \qquad \text{for all } s \text{ and } t$$

$$\frac{\partial^2 G_i}{\partial a_{st} \partial a_{uv}} = 0 \qquad \text{for all } s \ t \ u \text{ and } v$$

$$\frac{\partial^2 G_i}{\partial a_{st} \partial b_{wx}} = 0 \qquad \text{for all } a \ t \ w \text{ and } x$$

$$\frac{\partial^2 G_i}{\partial b_{wx} \partial b_{yz}} = 0 \qquad \text{for all } w \ x \ y \text{ and } z$$

$$\frac{\partial^2 G_i}{\partial a_{st} \partial x_k} = \delta_{is} \delta_{tk}$$

$$\frac{\partial^2 G_i}{\partial b_{wx} \partial x_s} = 0 \qquad \text{for all } w \ x \text{ and } s$$

$$\frac{\partial G_i}{\partial x_s} = a_{is}$$

The differential equations for the second partial derivatives are

$$\frac{d}{dt}\left(\frac{\partial x_i}{\partial a_{st} \partial a_{uv}}\right) = \sum_{k=1}^{n} a_{ik} \frac{\partial^2 x_k}{\partial a_{st} \partial a_{uv}} + \delta_{iu} \frac{\partial x_v}{\partial a_{st}} + \delta_{is} \frac{\partial x_t}{\partial a_{uv}}$$

$$\frac{d}{dt}\left(\frac{\partial^2 x_i}{\partial a_{st} \partial b_{wx}}\right) = \sum_{k=1}^{n} a_{ik} \frac{\partial^2 x_k}{\partial a_{st} \partial b_{wx}}$$

$$\frac{d}{dt}\left(\frac{\partial^2 x_i}{\partial b_{wx} \partial b_{yz}}\right) = \sum_{k=1}^{n} a_{ik} \frac{\partial^2 x_k}{\partial b_{wx} \partial b_{yz}} \qquad (26)$$

In all of the above expressions, i, s, t, u, v, w, and y run from 1 to n, x, and z from 1 to m

It is appropriate to make a remark about the order of magnitude of the task of calculating the second order partial derivatives needed in the Newton-Raphson method Assume n = 4 and m = 3 and assume 7 unknown parameters in the A and 7 unknowns in the B matrix Due to the symmetry

$$\frac{\partial^2 x_i}{\partial \alpha_j \partial \alpha_k} = \frac{\partial^2 x_i}{\partial \alpha_k \partial \alpha_j}$$

a total of $4*14*15/2 = 420$

second partial derivatives are required, which means that a system consisting of 480 (linear) differential equations (4 state variables, 56 first order partial derivatives and 420 second order partial derivatives) has to be solved Considering the simple form of the right hand sides of these equations, it is feasible to solve them on a digital computer with typically 32,000 words of core storage

The above derivation showed how the second partial derivatives of the state variables with respect to the unknown parameters can be found The necessary groundwork is now laid to consider the problem of determining stability derivatives using Newton's method

Let

$$z(\bar{\alpha}) = \sum_{i=1}^{n} \int_0^T \left[x_i(t,\bar{\alpha}) - \tilde{x}_i(t) \right]^2 dt \qquad (27)$$

Identify

$$\frac{\partial z}{\partial \alpha_j} \quad 2 \sum_{i=1}^{n} \int_0^T \left[x_i(t,\bar{\alpha}) - \tilde{x}_i(t) \right] \frac{\partial x_i}{\partial \alpha_j} \, dt \qquad (28)$$

with the function $H_j(\alpha)$ of the first paragraph

Let again $\bar{\alpha}^0$ be a point in the parameter space close to a local minimum of $z(\bar{\alpha})$ Then the n linear equations for obtaining the $\Delta\alpha_j$ are (see equation 23)

$$\sum_{k=1}^{U} \frac{\partial}{\partial \alpha_k} \left\{ 2 \sum_{i=1}^{n} \int_0^T \left[x_i(\bar{\alpha}^0,t) - \tilde{x}_i(t) \right] \frac{\partial x_i}{\partial \alpha_j} (\bar{\alpha}^0,t) \, dt \right\} \Delta\alpha_k$$

$$= - \sum_{i=1}^{n} \int_0^T \left[x_i(\bar{\alpha}^0,t) - \tilde{x}_i(t) \right] \frac{\partial x_i}{\partial \alpha_j} (\bar{\alpha}^0,t) \, dt \qquad (29)$$

$j=1 \quad U$

Carrying out the differentiation under the sum and integral sign and dropping the constant factor 2 gives

$$\sum_{k=} \sum_{i} \int \left\{ \frac{\partial x_i}{\partial \alpha_k} (\bar{\alpha}^0,t) \frac{\partial x_i}{\partial \alpha_j} (\bar{\alpha}^0,t) + \left[x_i(\bar{\alpha}^0,t) - \tilde{x}_i(t) \right] \frac{\partial^2 x_i}{\partial \alpha_k \partial \alpha_j} (\bar{\alpha}^0,t) \right.$$

$$\left. dt \right\} \Delta\alpha_k = - \sum_{i} \int_0^T \left[x_i(\bar{\alpha}^0,t) - \tilde{x}_i(t) \right] \frac{\partial x_i}{\partial \alpha_j} (\bar{\alpha}^0,t) \, dt$$

$j=1 \quad U \qquad (30)$

Knowing now

$$\frac{\partial^2 x_i}{\partial \alpha_k \partial \alpha_j} (\alpha,t),$$

the coefficients in the matrix for the linear equations can be obtained by an integration of known functions Also the elements of the right hand side vector can be obtained by an integration of known functions

The Modified Newton-Raphson Method

If the integrals in equation (30) are approximated by sums and if the second partial derivative term is neglected, the following equations are obtained

$$\sum_{k=1}^{U} \sum_{i=1}^{n} \sum_{\nu=1}^{M} \frac{\partial x_i}{\partial \alpha_k} (t_\nu) \frac{\partial x_i}{\partial \alpha_j} (t_\nu) \Delta\alpha_k =$$

$$\sum_{i=1}^{n} \sum_{\nu=1}^{M} \left[\tilde{x}_i(t_\nu) - x_i(t_\nu) \right] \frac{\partial x_i}{\partial \alpha_j} (t_\nu) \qquad (31)$$

for $j=1 \quad U$

Comparison of equation (31) with equations (18) shows that the two systems of equations for obtaining the corrections $\Delta\alpha_k$ are identical

This demonstrates that the application of the Newton-Raphson method to the minimization of the cost function (10) gives the same result as the method of quasilinearization if the second partial derivatives are neglected The possibility of modifying the Newton-Raphson method by neglecting the second partial derivatives was mentioned the first time by Balakrishnan in reference 8

DIRECT SEARCH BY EVOLUTIONARY PROGRAMMING

Basic Concepts of Evolutionary Programming

Evolutionary programming is described in detail in the book "Artificial Intelligence Through Simulated Evolution" by Fogel, Owens and Walsh (ref 3) In order that a reader unfamiliar with the concept of evolutionary programming might understand the direct search method described in the next section, a short summary of evolutionary programming is presented

Consider first a Moore-machine, a triplet (I,Σ,f), where I represents an input alphabet with a finite number of elements, Σ a set of states and f a transfer function from I x) to the set of states When an element of the input alphabet is received by a Moore-machine, it will transfer from one state to the next state in accordance with the rule laid down by the transfer function f The program used for the determination of stability derivatives is capable of handling Moore-machines with up to five states and with an input alphabet size of 60 Once the input alphabet and the number of states are specified, the transfer function f is given in tabular form specifying for each state and for each input symbol the next state reference

Evolutionary programming works essentially with finite state machines which can be described by a quintuplet (I,),f,O,g), where again I denotes an input alphabet, Σ a set of states, f the next state transfer function, O an output alphabet which may or may not be identical with I (in the application for stability derivatives, O contains only 6 different elements), and g an output function For a given output alphabet O the specification of g will uniquely determine a finite-state machine

Evolutionary programming is a method to find finite-state machines which will produce, for a given sequence of input symbols a sequence of output symbols which will minimize a certain cost function Evolutionary programming consists essentially of three basic procedures, an environmental comparison procedure, a mutation and selection procedure, and an output determination procedure

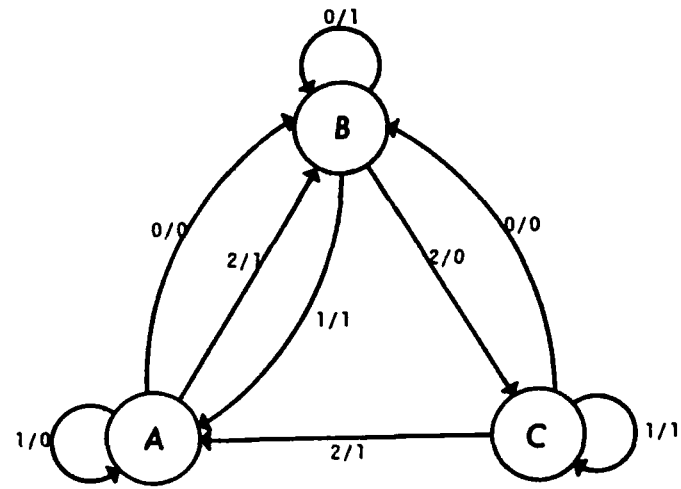


FIGURE 1

EXAMPLE OF A FINITE-STATE MACHINE

3 internal states, 3 symbol input alphabet, 2 symbol output alphabet

Figure 1 shows a finite-state machine with three states, three input symbols and two output symbols Assume that the finite-state machine is initially in state A and receives the input sequence 0 0 2 0 The following sequence of events will then take place

Present State	A	B	B	C	B	C	A
Input Symbol	0	0	2	0	2	2	1
Next State	B	B	C	B	C	A	A
Output Symbol	0	1	0	0	0	1	0

The output determination algorithm drives the Moore-machine with the given sequence of input symbols and determines those outputs (to each branch) that will minimize a given cost function The outputs are obtained in a deterministic manner

The mutation and selection procedure generates an offspring by randomly performing one of the following mutations of the present finite-state machine

Adding a state
Deleting a state
Changing a next state reference
Changing the start state

When a state is added, some next state references to previously existing states are changed randomly to connect the new state with the rest of the machine. The newly added state obtains as many next state references as there are input symbols These next state references are added randomly When a state is deleted, all the next state references which referred to that state are deleted and randomly connect to some other states

The two remaining mutations are self-explanatory The output determination routine determines now the proper output symbols to this mutated machine

A comparison is made whether the parent or the offspring obtains a lower value of the cost function and that finite-state machine yielding the lower value is kept as a new parent machine

As more and more information becomes available (larger and larger recall) over which the finite-state machines can be exercised, finite-state machines are generated which reflect, with increasing fidelity, the logic of the underlying process

Evolutionary Programming Applied to Function Minimization

The basic idea of evolutionary programming is to find finite-state machines which reflect in some sense the logic in the behavior of a system This may be an independent system or a system which interacts with its model and whose behavior, therefore, is dependent on the evolutionary program As an example of the first class, consider the problem of finding finite-state machines describing the logic of the changes in ocean temperature Clearly, the ocean temperature is independent of the logic found by the evolutionary program The problem of function optimization is an example of the second kind, such a process can be viewed as being a game between the evolutionary program and the cost function. The behavior of the value of the cost function in the past is now dependent on what "moves" the evolutionary program made, this is the interaction between the two "players" There is of course a clear distinction between the cost function and the "values of the cost function" The cost function itself is certainly independent of the optimization method used, but the "values of the cost function" depend on the path taken by the optimization procedure It may be mentioned here that Wilde, in his book on optimum seeking methods (ref 9) also talks about the "Opening Gambit" and the "End Game". The purpose of the evolutionary program in an optimum seeking procedure can be summarized as being the device which indicates which sequence of changes in the free parameters will be the most promising to reduce the value of the cost function This is done presently in the following way

Each free parameter can be changed in four different ways, in a positive or negative direction with either a large or small step size Consider now an input alphabet (α_i) consisting of four times as many symbols as there are unknown free parameters Each of these symbols represents a unique change in one of the parameters An evaluation of the cost function using this changed parameter will yield a new value of the cost function Designate the change in the cost function by $\bar{\beta}_1$, therefore, $\bar{\beta}_1 = z_1 - z_{1-1}$ Negative values of $\bar{\beta}$ correspond to improvements in the set of parameters, while a positive value means a degradation in the set of parameters Clearly, it is desirable to make $\bar{\beta}$ as small as possible (this corresponds to a large improvement) Define the follow-

ing intervals

$$a_1 < a_2 < 0 < a_3 < a_4$$

and define a corresponding output β where

$\beta = 1$ if $\bar{\beta} \leq a_1$

$\beta = 2$ if $a_1 < \bar{\beta} \leq a_2$

$\beta = 3$ if $a_2 < \bar{\beta} \leq 0$

$\beta = 4$ if $0 < \bar{\beta} \leq a_3$

$\beta = 5$ if $a_3 < \bar{\beta} \leq a_4$

$\beta = 6$ if $a_4 < \bar{\beta}$

For the given set of parameters there corresponds exactly one output symbol β to one input symbol α

Assume now that there exists a finite past history of pairs of (α_k, β_k) Clearly, the game of minimizing the cost function with respect to the given parameters consists now in choosing an α_{k+1} which will produce a β_{k+1} as small as possible It is here the evolutionary program comes into play Suppose that there is a finite-state machine which will "fit" the sequence (α_n, β_n), n = 1, 2 k Fit here means that if the finite-state machine is driven by the sequence of the α_n, it will produce the output sequence β_n Then at the k-th move, that finite-state machine is in a certain state, say S_j All possible inputs β will have a unique output associated with them It seems logical to assume that, since the finite-state machine was a perfect fit over the past, this finite-state machine contains information about the outcome associated with any given next input symbol Scanning all possible outcomes and searching for the lowest possible, the input symbol associated with the lowest output symbol can be determined Call the lowest possible output symbol β_{k+1}^{pred} The symbol associated with β_{k+1}^{pred} is now considered to be the evolutionary program's next move Note that several different input symbols may produce the same lowest value for the output symbol If this is the case, one among all the candidates for producing the lowest output symbol is chosen randomly for actual usage The parameter change associated with this symbol α_{k+1} is performed and a new evaluation of the payoff function occurs At this point, the two newly generated symbols α_{k+1} and β_{k+1}^{act} (which corresponds to the

actual change in the cost function), are added to the list of input/output pairs If the actual output, β_{k+1}^{act}, was a 3 or smaller, the game proceeds in the same way as described above, the recall being now one event longer If the output was 4 or more, the evolutionary program is considered as having made an action error. This case is described later in this section.

It is shown above how, in principle, a finite-state machine with a perfect fit over the past history is used as an "acting player" It is appropriate at this point to add some additional important details First, for any reasonable length of the sequence of the past moves, called the recall, it is unlikely that a finite-state machine with a perfect fit will be found For this, and other reasons, more than one finite-state machine are carried as possible players in the evolutionary program, specifically, for the problem solved here there are three finite state machines which are restricted in size. Machine 1 is a simple one-state machine while machines 2 and 3 can have any number of states between 2 and 5. Before a move is made, the "best" one of the three possible machines is selected as "player". Best means that machine with the lowest fit score The fit score is obtained in the following way. Given k pairs of input/output symbols, $\alpha_1, \beta_1, \alpha_2, \beta_2, \ldots \alpha_k, \beta_k$, drive the finite-state machine with the k inputs and for each move form the difference $|\beta_m - \beta_a|$, where β_m is the output predicted by the machine and β_a the output that actually occured Divide the sum $\sum_{j=1}^{k} |\beta_m^j - \beta_a^j|$ by k and define this quantity as the fit score If the actual output β_{k+1}^{act} is greater than or equal to three (a degradation in the cost function) the evolutionary program is considered to have made an error or a "bad action" has occurred Machines 2 and 3 are now mutated Mutation means that with probability

p_1 a start state is changed

p_2 a next state reference is changed

p_3 a state is added

p_4 a state is deleted

where, of course

$$\sum_{i=1}^{4} p_i = 1$$

If the mutant has already the maximum number of states, p_3 automatically is set to zero and analogously p_4 equals zero if the machine has only 2 states. Machine 1 is not mutated, because the only possible mutation would be to add a state, but the intention is to keep machine 1 a one state machine.

The offspring machine is driven over the recall and its fit score is evaluated. If the fit score is worse than that of the parent machine, the offspring is replaced by the parent and another mutation is tried. A parameter in the program limits the number of trials for obtaining a better machine. After machines 2 and 3 have been mutated (or at least an attempt has been made to mutate them), the machine with the best fit score of the three machines is chosen as the actor and the game proceeds in its normal way.

Two important details of the evolutionary program have not yet been discussed, the setting of the outputs and the treatment of the unexercised state-transitions. First note that the mutations affect only the structure of the finite-state machines but not their outputs. It is clear that for a finite-state machine with a given structure and a given start state, there exists at least one setting of the outputs which will minimize the fit score. Assume first that no state transition is exercised more than once during the recall. Then, each one of the exercised state-transistions is assigned a unique output symbol (namely the one corresponding to the actual occurred output symbol during the transition) and the fit score will obviously be zero, because for all j, $j=1 \quad k$, $\beta_a^j = \beta_m^j$. More important is the case where some state-transitions are exercised more than once during the recall. It is possible that, in order to fit the actual data, different symbols would be required each time the transition is exercised. In such a case, the output symbol is set to some weighted average (rounded to the nearest integer) of the desired output symbols. Also a new fit score is defined, which is equal to the fit score as described above divided by the number of times that non-unique state-transitions have occurred, this is called the normalized fit score and it is this normalized fit score on which the choice

of the acting player is based. The third possibility is that a state-transition is never exercised during the recall and therefore the output symbol associated with this state-transition has no influence on the fit score. It proved to be advantageous from a programming point of view to assign a special symbol to those outputs. An output of zero now designates an unset output value.

A last point to discuss is the procedure used if analyzing the acting player, a move that will produce an improvement in the cost function cannot be found, or expressed in the alphabet of the evolutionary program, if in a given state no input symbol will produce a predicted β of 3 or better. If this occurs, the evolutionary program is not used to generate a symbol for the next move. The next move is obtained by scanning the past moves and finding the most recent move which produced a β of 3 or less. If at any move, an actual output of 4 or greater is generated, the evolutionary program is said to have made an action error. If this occurs, the next move or next moves will not be determined by the evolutionary program, but rather by a subprogram which essentially tries out whether a step in the reverse direction is better and it keeps trying until it again finds a successful move, always restoring the coefficients to their old value after an unsuccessful move. Details about this subroutine can be obtained from the flow chart in Figure 2. This subprogram also guarantees that at the end, a local minimum of the cost function within the specified levels of changes in the parameters has been found, because only after an exhaustive unsuccessful search over all possible single changes in the parameters is the run terminated. A second way to terminate the run is by limiting the number of moves. After termination, the final value of the coefficients are printed together with the complete sequence of input/output pairs. Since the evolutionary program requires some environment in the past, initially, to start the program, a separate subroutine generates a prescribed number of input symbols and the corresponding output symbols are calculated. This is called *the initial environment*. *The moves which generate* the initial environment are performed in a stochastic manner.

Summarized, the features of the present version of the evolutionary program for the determination of stability and control derivatives are as follows.

(text continued on page 37)

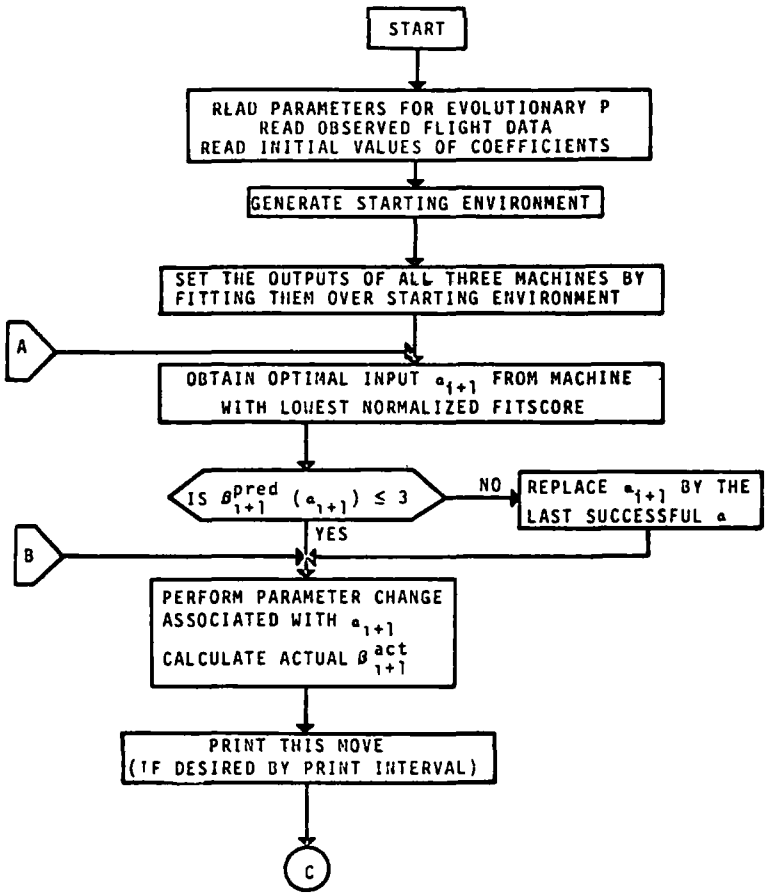


FIGURE 2

FLOWCHART OF EVOLUTIONARY PROGRAM FOR FUNCTION MINIMIZATION

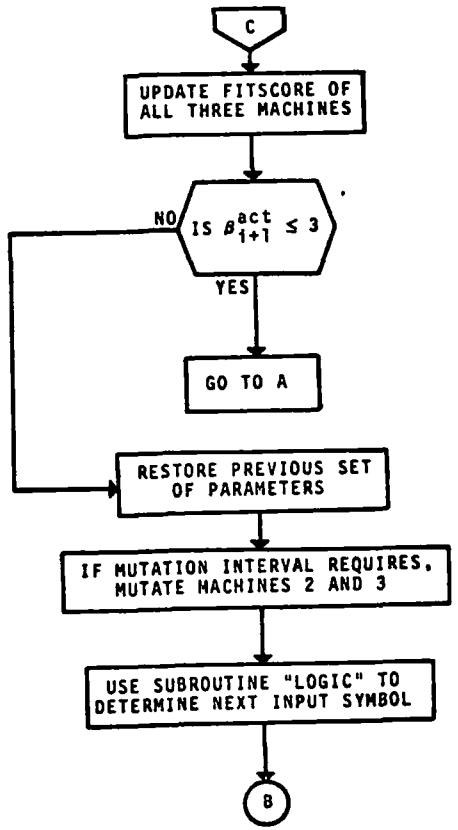


FIGURE 2 (CONTINUED)

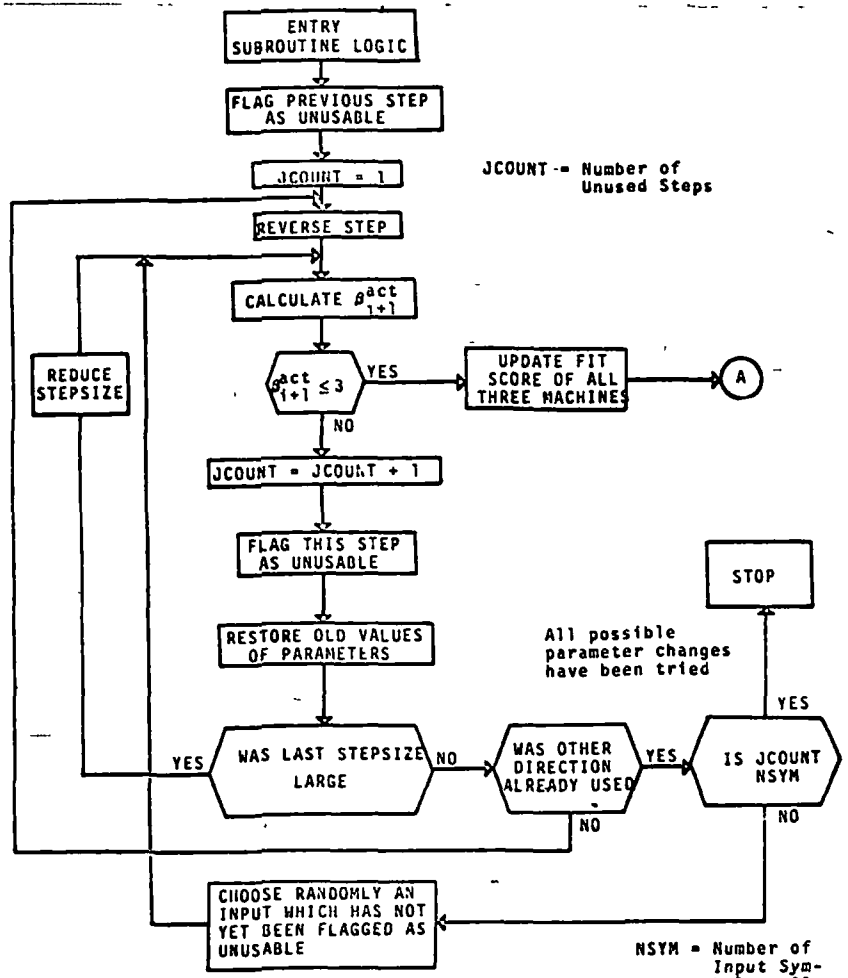


FIGURE 2 (CONCLUDED)

Differential Equation $\dot{x} = Ax + Bu$ where

A is of dimension (4,4)
B is of dimension (4,3)

Cost Function:

$$z = \sum_{i=1}^{4} w_i^2 \int_0^T (x_i - \tilde{x}_i)^2 \, dt$$

Free Parameters a_{11} a_{12} a_{13} a_{21} a_{22} a_{23} a_{31} a_{33}

b_{11} b_{12} b_{13} b_{21} b_{22} b_{23} b_{33}

Number of Finite-State Machines:
1 one-state machine
2 2 to 5 state machines

Input Symbols 60
Output Symbols 6

Input to the Program

For each machine
Initial configuration
Initial start state
Maximum recall length

Probability distribution for the 4 types of mutation

p_1 = probability of changing start state

p_2 = probability of changing next state reference

p_3 = probability of adding a state

p_4 = probability of deleting a state

Maximum number of moves
Print interval.
Number of errors allowed before a mutation occurs
Maximum number of machines tried at a mutation
For all 60 input symbols the change in the coefficient associated with this input symbol
The interval limits for the determination of the output symbol, a_i, i=1 . 4.
The initial values of the 28 elements of the A and B matrix.
The observed time histories
The integration step size

Results

Some key results are listed below for a typical run of the evolutionary program on the CDC 3600 computer

Running time		6 minutes
Initial environment length	50	
Total number of moves of the evolutionary program	400	
Total number of function evaluations	750	
Value of z		
	initially	1 811
	after init 50 moves	1 038
	after 100 moves of Evol Pr	0 564
	after 200 moves	0 491
	after 400 moves	0 463

Input Symbols The 60 symbols of the input alphabet represent changes of

+2%, -2%, +0 2%, -0 2%

of the coefficients

a_{11}, a_{12}, a_{13}, a_{21}, a_{22}, a_{23}, a_{31}, a_{33}

b_{11}, b_{12}, b_{13}, b_{21}, b_{22}, b_{23}, b_{33}

in this order

Intervals for output symbols

Output Symbol	Corresponds to
1	$\bar{R} \leq -5\ 0\ 10^{-3}$
2	$\bar{R} \leq -5\ 0\ 10^{-4}$
3	$\bar{R} \leq 0$
4	$\bar{R} \leq 1\ 0\ 10^{-3}$
5	$\bar{R} \leq 1\ 0\ 10^{-2}$
6	$\bar{R} > 1\ 0\ 10^{-2}$

The initial coefficients were those found by the least squares procedure and are given in the following two matrices

A

-0 0999	0 595	-22 48	0
0 00641	0,0668	1 036	0
0 1147	-1	0 0238	0 00698
1	0	0	0

B

12 99	15 11	0.361
0 487	-1.764	0.00731
0	0	-0.00272
0	0	0

Note that in this experiment a_{31} was considered as one of the unknown parameters.

The coefficients at the end of the run were

A

-0 153	0.163	-21.67	0
0.00615	0.0642	1 108	0
1.216	-1.	-0.0190	0.00698
1	0	0	0

B

13 25	14 53	0 378
0.551	2 06	-0.00221
0	0	-0.00261
0	0	0

Note that the first eleven moves of the evolutionary program (move 51 through 61) produced all outputs of 1, which is quite remarkable, considering that the longest string of "1" in the first fifty moves was only of length 3 (see Table I)

Although the optimization method using the evolutionary program works satisfactorily in its present form, there exist possibilities to improve its performance. A first improvement consists in preventing the evolutionary program from getting "trapped" in a long string of input symbols which all produce an output 3 (a very slight improvement in the cost function) If unlimited computer time were available, these long strings of outputs of 3 would be all right, but in the interest of of saving computer time, an attempt should be made to find parameter changes which will improve the cost function more rapidly. Such values may be found more quickly if after a string of outputs 3 with some given fixed length, a random

(text continued on page 42)

TABLE I - INPUT - OUTPUT HISTORY OF THE FIRST 150 MOVES OF THE EVOLUTIONARY PROGRAM

MOVE NUMBER	INPUT SYMBOL	OUTPUT SYMBOL (a)
1	37	1
2	31	3
3	16	2
4	51	6
5	52	1
6	35	5
7	36	2
8	28	5
9	27	2
10	11	6
11	12	1
12	8	3
13	3	3
14	22	6
15	21	1
16	10	1
17	16	2
18	48	2
19	19	5
20	20	2
21	38	6
22	37	1
23	59	3
24	36	2
25	25	1
26	2	5
27	1	2
28	25	1
29	54	1
30	48	2
31	49	6
32	50	1
33	1	2
34	7	4
35	8	3
36	53	6
37	54	1
38	25	1
39	52	1
40	42	6
41	41	1
42	26	6
43	25	2
44	56	2
45	24	6
46	23	1
47	25	6
48	26	4
49	27	4
50	28	3
51	54	1 *
52	12	1 *
53	52	1 *
54	54	1 *
55	52	1 *
56	54	1 *
57	54	1 *
58	52	1 *
59	54	1 *
60	23	1 *
61	10	1 *
62	52	2 *
63	41	1 *
64	12	4 *
65	11	4
66	37	2 *
67	52	4 *
68	51	2
69	52	4 *
70	51	2
71	54	1 *
72	54	2 *
73	23	3 *
74	37	6 *
75	38	1
76	41	5 *

[a] An asterik after the output symbol indicates that this move was determined by the evolutionary program

TABLE I (CONCLUDED)

MOVE NUMBER	INPUT SYMBOL	OUTPUT SYMBOL (a)
77	42	6
78	43	3
79	23	2 *
80	41	6 *
81	42	5
82	43	4
83	44	4
84	10	6 *
85	9	5 *
86	11	2 *
87	54	2 *
88	54	4 *
89	53	5
90	55	4
91	56	3
92	54	5 *
93	53	5
94	55	4
95	56	3
96	41	6 *
97	42	5
98	43	4
99	44	2
100	50	6 *
101	49	1
102	52	4 *
103	51	3
104	54	1 *
105	52	5 *
106	51	2
107	21	2 *
108	38	1 *
109	38	2 *
110	54	2 *
111	10	6 *
112	9	6
113	11	2
114	49	1 *
115	49	6 *
116	50	6
117	51	4
118	52	3
119	49	6 *
120	50	6
121	51	4
122	52	4
123	38	5 *
124	37	2 *
125	38	5 *
126	37	5
127	39	4
128	40	4
129	1	2 *
130	12	4 *
131	11	4
132	48	4 *
133	47	2 *
134	10	6 *
135	9	6
136	11	4
137	12	4
138	49	5 *
139	50	6 *
140	51	2 *
141	54	1 *
142	54	2 *
143	54	5 *
144	53	5
145	55	3
146	54	5 *
147	53	5
148	55	4
149	36	4
150	38	2 *

[a] An asterik after the output symbol indicates that this move was determined by the evolutionary program

search procedure similar to the one used to generate the initial starting environment, is used for a given number of moves

A second improvement would, as the search procedure approaches the minimum of the cost function, automatically change the magnitude of the changes in the coefficients (say reduce them by a factor of 10) and also reduce the values of $|a_1|$ through $|a_4|$ This latter change would increase the sensitivity of the evolutionary program to changes in the cost function

EXPERIMENTAL RESULTS AND COMPARISON OF THE TWO METHODS

Two computer programs were developed, one implementing the quasilinearization method and the other one the direct search method using evolutionary programming

The first experimental runs with these programs were made with data which did not originate from actual flight tests, but which were obtained from solving four simultaneous differential equations of form (1) on a hybrid computer and using the measured and digitized data from these runs. Clearly, since these data originated from a process described exactly by a differential equation of the form considered here, and since the only errors were roundoff errors in the digitized data, the coefficients were found quite accurately and the observed time histories were matched by the calculated time histories with the same accuracy as the originally given time history data (three to four significant digits).

The next case analysed the flight test data of an X-15 flight. Measured data of p, r, β, ϕ, and of $\dot{p}$ and $\dot{r}$ were available at 0.025 second intervals for a total observation time of 6 seconds. For the calculation, every second point of these time histories was used.

A first approximation to the unknown coefficients was obtained using the least squares method In the experiments with the program using the quasilinearization method, for the element a_{31} the observed angle of attack has been used.

By the least squares method the following parameters and estimates of their variances were obtained.

A

-0.101±0 011	0 539±0.213	-22.43±0 15	0
0 0064±0 001	0 0619±0.019	1.036±0.01	0
0 114±0 0019	-1	-0.058±0.021	0 00698
1	0	0	0

B

12 99±0 40	15 15±0.38	0.359±0.006
0 498±0 035	-1 760±0.034	-0.0074±0.00058
0	0	0.0148±0.0013
0	0	0

After five iterations with the quasilinearization method, the following coefficients and error estimates were obtained (see also Figure 8 for comparison)

A

-0 191±0 04	2 853±0 75	-24 08±0 24	0
0 0041±0 0028	-0 126±0 06	0 974±0 025	0
u(t)	-1	- 0 020±0 035	0 00698
1	0	0	0

B

14 21±1 45	19.37±1 72	0 406±0 025
0 709±0 128	-1.951±0 159	-0 002±0 002
0	0	-0 0012±0 0008
0	0	0

It was beyond the scope of the work performed under this contract to develop methods for obtaining error bounds on the calculated stability and control derivatives Nevertheless, the methods used to find numerical values for the variances in the calculated derivatives seem quite reasonable and they allow at least an estimate of the expected relative accuracy of the parameters. For instance, looking at the value of $a_{12}(L_r)$ and its estimated variance in the pure least squares solution indicates that this parameter was determined with very little accuracy Indeed, the a_{12} found after the fifth iteration differs from the a_{12} from the least squares solution by a factor of about five, and again, the estimate of the error after the fifth iteration is still fairly large On the other hand, looking at $a_{23}(N_\beta)$ the relative small variance in the least squares solution is an indication that this parameter can be determined relatively accurately and the final value of a_{23} after five iterations differs only about 6% from the value found by the pure least square procedure

Figures 3 through 6 show the observed roll and yaw rates and the sideslip and bank angles On the same graphs are shown the time histories obtained using the coefficients of the pure least squares solution and the trajectories obtained with the coefficients after five iterations of the combined gradient-least squares method Figure 7 shows the corresponding aileron and rudder deflections

(text continued on page 51)

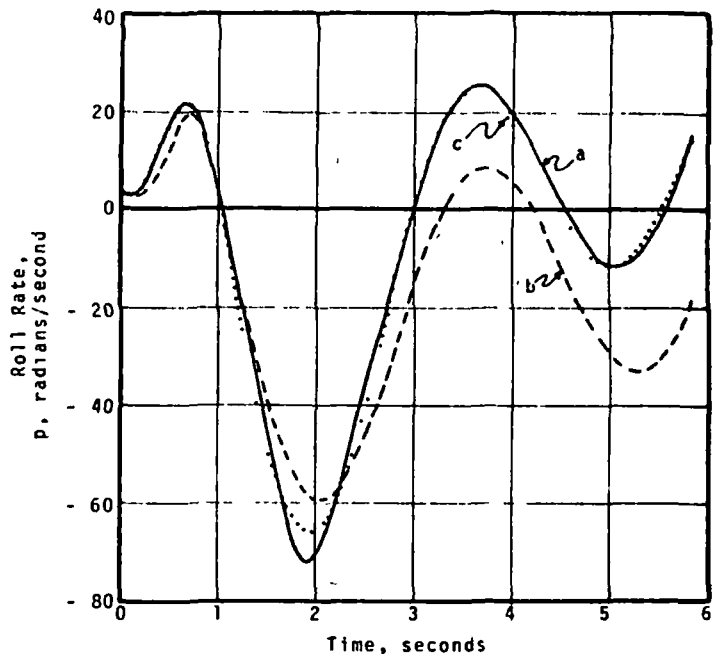


FIGURE 3.
OBSERVED AND CALCULATED ROLL RATES OF X-15 FLIGHT TEST.

a - Observed roll rate
b - Calculated roll rate using coefficients found with pure least squares procedure
c - Calculated yaw rate using coefficients found with 5 iterations of the quasilinearization method.

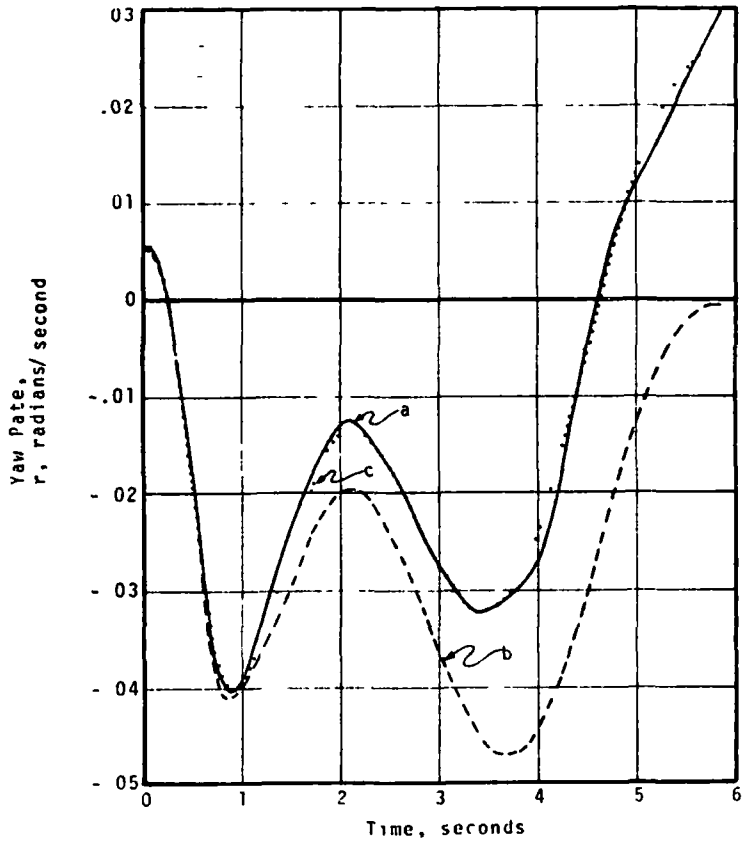


FIGURE 4

OBSERVED AND CALCULATED YAW RATES OF X-15 FLIGHT TEST

a - Observed yaw rate
b - Calculated yaw rate using coefficients found with pure least squares procedure
c - Calculated yaw rate using coefficients found with 5 iterations of the quasilinearization method

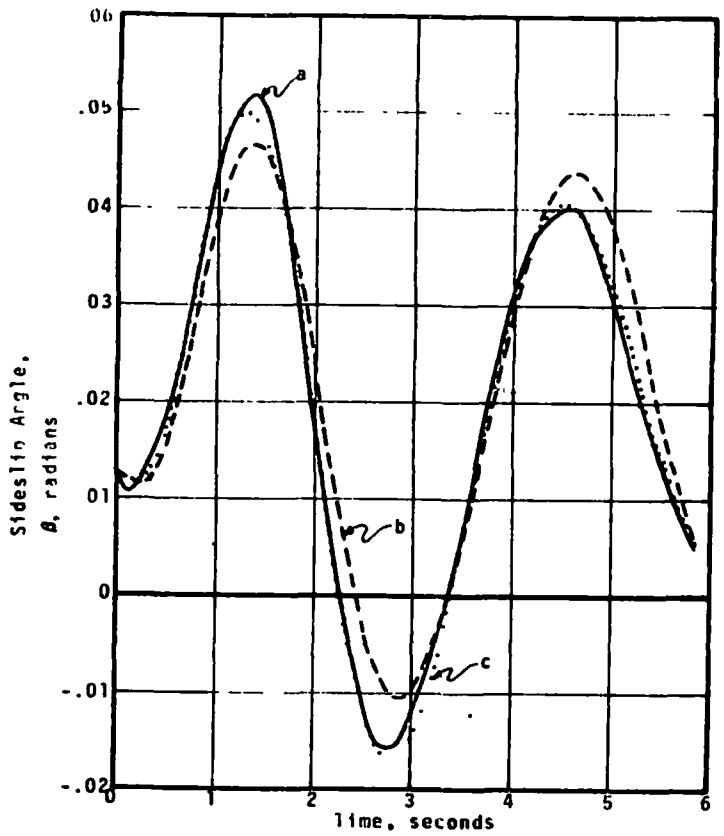


FIGURE 5.
OBSERVED AND CALCULATED SIDESLIP ANGLE OF X-15 FLIGHT TEST.

a - Observed sideslip angle.
b - Calculated sideslip angle using coefficients found with pure least squares procedure.
c - Calculated sideslip angle using coefficients found with 5 iterations of thequasilinearization method.

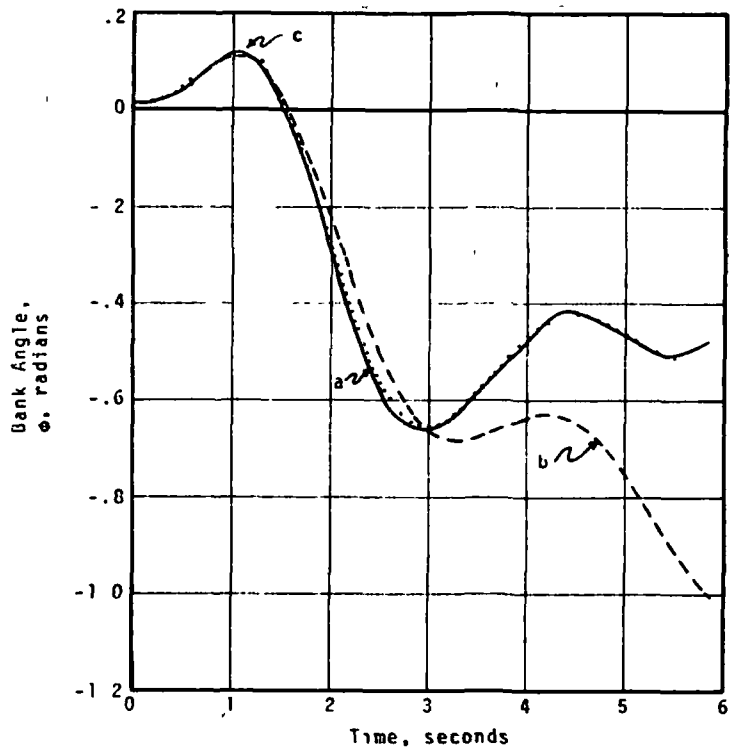


FIGURE 6
OBSERVED AND CALCULATED BANK ANGLE OF X-15 FLIGHT TEST

a - Observed bank angle
b - Calculated bank angle using coefficients found with pure least squares procedure
c - Calculated bank angle using coefficients found with 5 iterations of the quasilinearization method

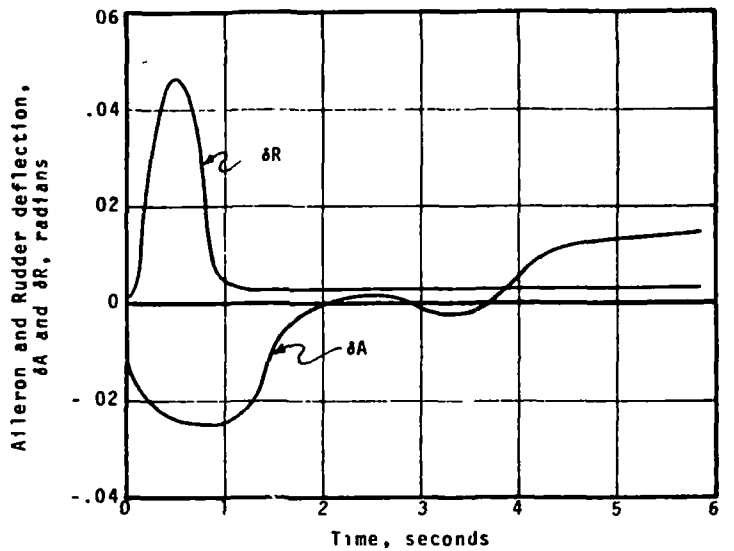
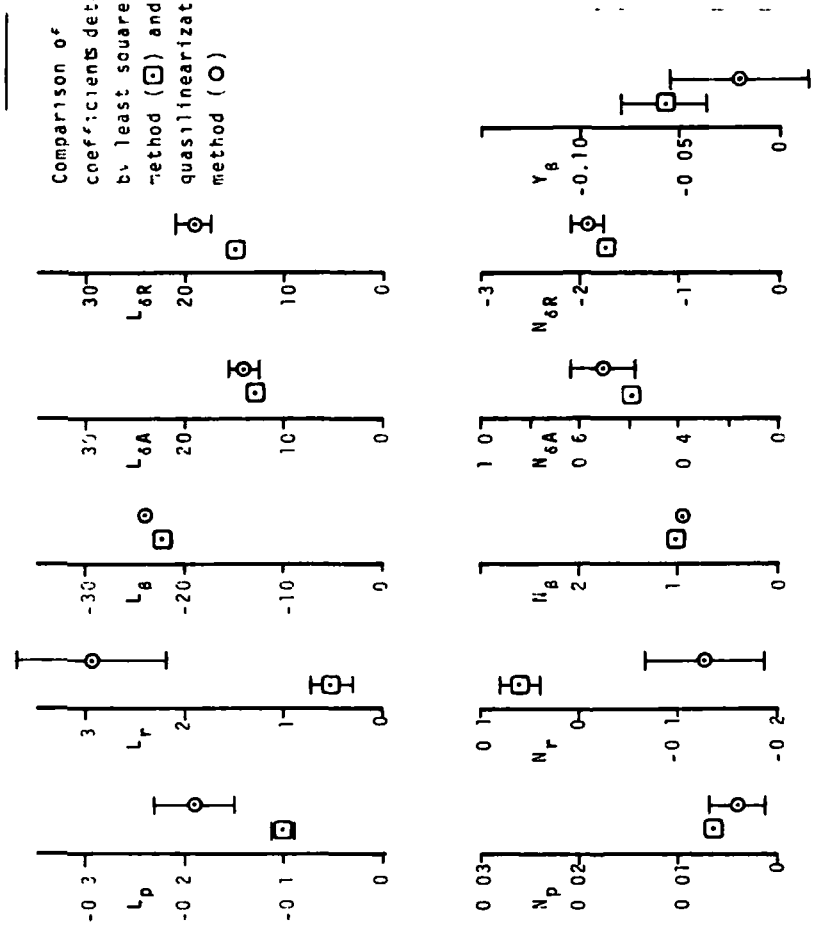


FIGURE 7
AILERON AND RUDDER DEFLECTION OF X-15 FLIGHT TEST

δA - Aileron deflection.
δR - Rudder deflection

FIGURE 8



The experimental results of the evolutionary program were discussed in the section "Direct Search By Evolutionary Programming".

Figure 8 compares the values of the coefficients obtained by the least squares method with those obtained by quasilinearization, as well as the corresponding estimates of the variances The fact that the estimated variance in the least squares method is smaller than in the quasilinearization method should not be taken as an indication that the least squares coefficients are closer to the true values than the ones obtained by the quasilinearization method It seems that the parameter estimates obtained by the least squares method are not unbiased, and as pointed out earlier, the estimates of the variances, as developed in this report merely indicate the relative accuracy between the different coefficients

At this point, it seems appropriate to compare the performance of the two methods. Judging only according to efficiency in computer time, the method of quasilinearization is clearly superior to evolutionary programming for a problem with linear differential equations and quadratic cost function. In about one minute computer time (CDC 3600) five iterations on a time history with about 150 measured points can be performed These five iterations yield a set of coefficients which minimize the cost function. The coefficients are accurate to four to five significant figures and as a by-product, the estimates of the variances are obtained

On the other hand, a six minute run on the same computer using the evolutionary programming technique yielded a final value of the cost function still about twice the size of the true minimum value Furthermore, no estimates of the error bounds are available with this method The evolutionary programming technique of minimizing functions with a relatively large number of unknowns may have advantages in system identification problems where a nonlinear model of the plant is required or where a nonquadratic error criterion has to be used The combination of a general numberical integration technique (such as for instance Runge-Kutta) and evolutionary programming allows quick changes both in the differential equations of the model and of the form of the cost function

CONCLUSIONS AND RECOMMENDATIONS

The results of this report show that the method of quasilinearization results in an efficient digital computer program which allows determining those values of the stability and control derivatives which minimize the integral of the weighted squared difference between the observed time history and the one obtained by solving the differential equations using the observed control variables and the parameters to be determined. A few iterations (typically three to five) will yield the correct values (the ones which minimize the cost function) of the unknowns and estimates of their variances can be obtained

The second method which is based on evolutionary programming cannot compete successfully in the case of linear differential equations and a quadratic error function, but it may have advantages in nonlinear process identification problems

The fact that a method is available that solves the minimization problem of a given cost function for a given form of the systems' differential equations (here linear) should not lead to the conclusion that the problem of determining stability and control derivatives from flight data is solved in its widest engineering sense A number of important questions are still open, for instance:

(1) How does the inclusion or omission of a fit of the observed yaw and roll rate derivatives influence the coefficients and their variances?

(2) What criterion should be applied in choosing the weighting factors?

(3) Is it possible to give variances which are theoretically more solidly founded and which distinguish between errors due to measurement noise and inadequacy of the mathematical model?

The analysis of the X-15 flight data shown in this report seems to indicate that the assumed mathematical model may not be quite adequate Especially the fit to the roll rate suggests that there are certain terms missing in the roll moment equation, these might be unsteady flow derivatives

The experiments suggest that additional work on the choice of the mathematical model with alternative forms of the equations of motion (possibly nonlinear equations) be performed Experiments, where stability and control derivatives obtained from

one flight test may indicate which mathematical models give the most consistent results and therefore are the most realistic ones Computational methods and computer programs are now available which may help to advance the state of the art in the determination and possibly the usage of stability and control derivatives

Decision Science, Inc
San Diego, September 25, 1968.

REFERENCES

1. Taylor, Lawrence W., and Iliff, Kenneth W A Modified Newton-Raphson Method for Determining Stability Derivatives from Flight Data. Paper presented at the 2nd International Conference on Computing Methods in Optimization Problems, San Remo, Italy, September 9-13, 1968.

2. Young, Peter C Regression Analysis and Process Parameter Estimation. .A Cautionary Message Simulation, Vol. 10, No. 3, March 1968. Pages 125-128.

3. Fogel, Lawrence J , Owens, A. J., and Walsh, M. J · Artificial Intelligence Through Simulated Evolution, John Wiley and Sons, Inc. 1966

4. Howard, S The Determination of Lateral Stability and Control Derivatives from Flight Data. Canadian Aeronautics and Space Journal, March 1967. Pages 127-134.

5. Smith, Gene A. The Theory and Applications of Least Squares. NASA TM X-63127, 1967.

6 Etkin, B. Dynamics of Flight Stability and Control. John Wiley and Sons, July 1959

7. Reinsch, Christian H.. Smoothing by Spline Functions. Numerische Mathematik 10, 1967. Pages 177-183.

8 Balakrishnan, A. V. Communication Theory McGraw-Hill Book Company, Inc 1968

9. Wilde, D J Optimum Seeking Methods. Prentice-Hall, Inc. 1967

10 Anon Dynamics of the Airframe Report AE-61-4II, Northrop Corporation, Norair Division, September, 1952.